SHGetFileInfo

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-16

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 7610 bytes

Attack Category	• Path spoofing or confusion problem
Vulnerability Category	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• File Management
Location	• shlobj.h
Description	Retrieves information about an object in the file system, such as a file, a folder, a directory, or a drive root. SHGetFileInfo is vulnerable to TOCTOU attacks. This function could be a check or a use-category call within the scope of TOCTOU parlance.
APIs	**Function Name** / **Comments** SHGetFileInfo / check
Method of Attack	The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results.
Exception Criteria	None noted
Solutions	**Solution Applicability** / **Solution Description** / **Solution Efficacy** Generally applicable. / Utilize a file descriptor version of check and use functions, if possible. / Effective

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

	Generally applicable.	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.
	Generally applicable.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.
Signature Details	DWORD_PTR SHGetFileInfo(LPCTSTR pszPath, DWORD dwFileAttributes, SHFILEINFO *psfi, UINT cbFileInfo, UINT uFlags);		
Examples of Incorrect Code	`/* Check occurs */` `#include "sys/types.h"`		

```
#include "sys/stat.h"

void OnTest( HWND hWnd )
{
OPENFILENAME ofn;
char szFileName[MAX_PATH] = "/
home/cnd/mod_done";
int check_status;
struct stat statbuf;

check_status=stat(szFileName,
&statbuf);

ZeroMemory( (LPVOID)&ofn,
sizeof(OPENFILENAME) );
ofn.lStructSize =
sizeof(OPENFILENAME);
ofn.hwndOwner = hWnd;
ofn.lpstrFileTitle = szFileName;
ofn.nMaxFileTitle = MAX_PATH;
ofn.Flags = OFN_FILEMUSTEXIST |
OFN_HIDEREADONLY | OFN_EXPLORER;

if( GetOpenFileName( &ofn ) )
{
SHFILEINFO shfi;

SHGetFileInfo( (szFileName,
FILE_ATTRIBUTE_ARCHIVE, &shfi,
sizeof(SHFILEINFO), SHGFI_ICON |
SHGFI_LARGEICON );
```

Examples of Corrected Code

```
begin
// This examples assumes a
TImageList component has already
been associated
// with the SmallImages property.
ListView1.SmallImages.Handle :=
SHGetFileInfo('', 0, Dummy,
SizeOf(Dummy),
SHGFI_SYSICONINDEX or
SHGFI_SMALLICON);

void OnTest( HWND hWnd )
{
OPENFILENAME ofn;
char szFileName[MAX_PATH];

szFileName[0] = '\0';

ZeroMemory( (LPVOID)&ofn,
sizeof(OPENFILENAME) );
ofn.lStructSize =
sizeof(OPENFILENAME);
ofn.hwndOwner = hWnd;
ofn.lpstrFileTitle = szFileName;
ofn.nMaxFileTitle = MAX_PATH;
```

	`` ofn.Flags = OFN_FILEMUSTEXIST | OFN_HIDEREADONLY | OFN_EXPLORER; if(GetOpenFileName(&ofn)) { SHFILEINFO shfi; SHGetFileInfo(szFileName, FILE_ATTRIBUTE_ARCHIVE, &shfi, sizeof(SHFILEINFO), SHGFI_ICON | SHGFI_LARGEICON); ```
Source References	• http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wceui40/html/cerefshgetfileinfo.asp[2] • Stowers, Brad. Image Is Everything[3] (1997). • http://www.expertmg.co.jp/html/cti/vctips/icon.htm
Recommended Resource	
Discriminant Set	**Operating System**: • Windows **Languages**: • C • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com